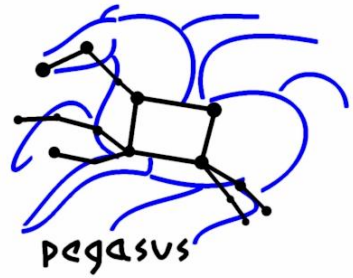


Automating processing with workflow environments

Ewa Deelman

Center for Grid Technologies

USC Information Sciences Institute

Workflow

- Complex applications can be viewed as workflows
  - Workflow activities (nodes) represent application components
  - Workflow dependencies represent the interactions between the components
- A workflow can be viewed as a recipe of how to produce a given dataset
- An executed workflow can carry with it provenance information for the dataset: what application components and resources were used
  - The workflow can include preservation processes that are applied to each digital record.

Scientific "Workflows" vs Business Workflows

- **Business Workflows (BPEL4WS* …)**
  - Task-orientation: *travel reservations; credit approval; BPM; …*
  - Tasks, documents, etc. undergo modifications (e.g., flight reservation from *reserved* to *ticketed*), but modified WF objects still identifiable throughout
  - Complex control flow, complex process composition (danger of control flow/dataflow "spaghetti")

  ➔ Dataflow and control-flow are often ***separated***
- **Scientific "Workflows"**
  - Dataflow and data transformations
  - Data problems: volume, complexity, heterogeneity
  - Grid-aspects
    - > Distributed computation
    - > Distributed data
  - User-interactions/WF steering
  - Data, tool, and analysis integration

  ➔ Dataflow and control-flow are often ***integrated***

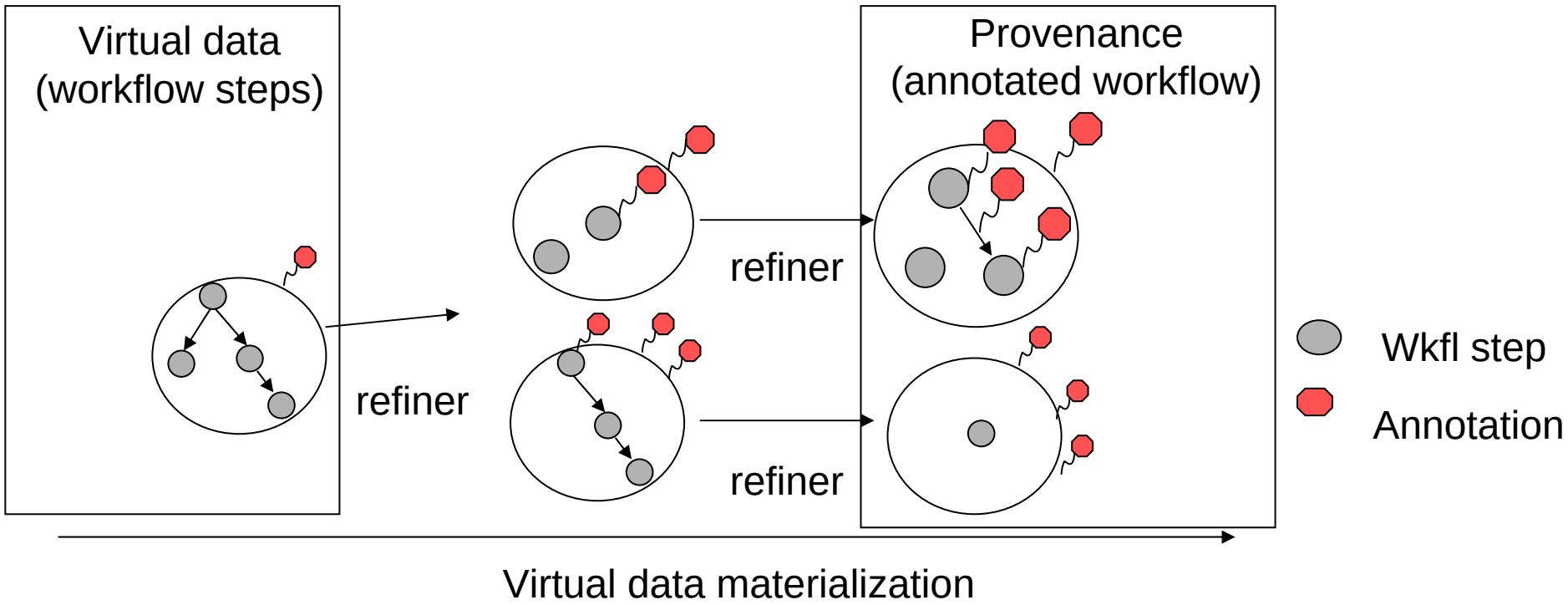
***Business Process Execution Language for Web Services** (in case you wondered)

Scientific "Workflows": Some Findings

- More dataflow than (business control-/) workflow
  - DiscoveryNet, Kepler, SCIRun, Scitegic, Triana, Taverna, …,
- Need for "programming extensions"
  - Iterations over lists (foreach); filtering; functional composition; generic & higher-order operations (zip, map(f), …)
- Need for abstraction and nested workflows
- Need for data transformations (WS1➔DT➔WS2)
- Need for rich user interaction & workflow steering:
  - pause / revise / resume
  - select & branch; e.g., web browser capability at specific steps as part of a coordinated SWF
- Need for high-throughput data transfers and CPU cyles: "(Data-)Grid-enabling", "streaming"
- Need for persistence of intermediate products and provenance

Relationship between virtual data, and provenance

- Virtual data can be described by a workflow, that undergoes an refinement process to obtain a workflow in the “done” state
- The refinement process can generate provenance information that describes how the workflow was executed



Annotations can be used to describe the state of the workflow, or provide additional provenance information

GriPhyN Software to support workflow evolution

- Workflow Generation: how do you describe the workflow (at various levels of abstraction)? (Chimera)
- Workflow Mapping/Refinement: how do you map an abstract workflow representation to an executable form? (Pegasus)
- Workflow Execution: how do you reliably execute the workflow? (Condor's DAGMan)
- Workflow Provenance: how do you record all information about the newly created data? (Chimera)

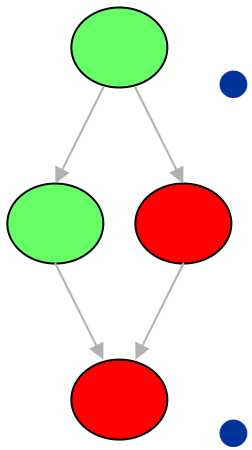
Chimera's VDL: Virtual Data Language Describes Data Transformations

- Transformation
  - Abstract template of program invocation
  - Similar to "function definition"
- Derivation
  - "Function call" to a Transformation
  - Store past and future:
    - A record of how data products were generated
    - A recipe of how data products can be generated
- Invocation
  - Record of a Derivation execution
- Developed at ANL and UofC

Pegasus: Planning for Execution in Grids

- Maps from abstract to concrete workflow
  - Algorithmic and AI-based techniques
- Shields from the Grid details
- Finds appropriate resources to execute
- Reuses existing data products where applicable
- Publishes newly derived data products
  - Chimera virtual data catalog
  - Provides provenance information
- Can run the workflow on a variety of resources and across computing platforms
- Can opportunistically take advantage of available resources (through dynamic workflow mapping)
- Developed at ISI

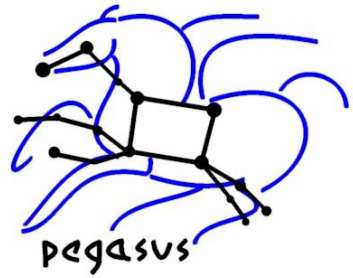
Condor's DAGMan



- Developed at UW Madison (Livny)
- Executes a concrete workflow
- Makes sure the dependencies are followed
- Executes the jobs specified in the workflow
  - Execution
  - Data movement
  - Catalog updates
- Provides a "rescue DAG" in case of failure

For more information

- GriPhyN project www.griphyn.org
- Chimera www.griphyn.org/chimera
- Pegasus pegasus.isi.edu



Additional slides

Workflow Evolution

- Workflow description
  - Provenance metadata
  - Partial, abstract description of the workflow segments
  - Full, abstract description of the entire workflow
  - A concrete, executable workflow that defines which compute resources are used
- Workflow refinement
  - Take a description and produce an executable workflow
- Workflow execution
  - Process the steps of the workflow on the grid

Benefits of the workflow approach

- The workflow exposes
  - the structure of the application
  - maximum parallelism of the application
- Workflow refiners shield users from execution environment details
- Workflow refiners can take advantage of the structure to
  - Set a planning horizon (how far into the workflow to plan)
  - Cluster a set of workflow nodes to be executed as one (in case the nodes has a small computational granularity)
- Workflow refiners can improve the performance of the application by customizing the mapping to particular resources (Teragrid vs condor pools)
- Workflow refiners and executors can record all the execution steps and thus provide provenance information necessary to evaluate the quality of the data

Workflow Refinement

- The workflow can undergo an arbitrarily complex set of refinements
- A refiner can modify part of the workflow or the entire workflow
- A refiner uses a set of Grid information services and catalogs to perform the refinement (metadata catalog, virtual data catalog, replica location services, monitoring and discovery services, etc.)

Workflow Refinement and execution